ISD Checklist

Number: 580-CK-009-01
Effective Date: August 1, 2004
Expiration Date: August 1, 2009

Approved By: (signature)
Name: Joe Hennessy
Title: Chief, ISD

Asset Owner: GSFC Engineering Process Group
Title: Software Contents of the Mission-Level CDR

Asset Type: Checklist
PAL Number: 2.3.2.4

Software Contents of the Mission-Level Critical Design Review (CDR)

Use this checklist to ensure that key elements of the software project's detailed design, development and management activities, and status are presented for review as part of the System (Mission-Level) CDR.

- ❑ **Management overview,** highlighting changes since Code 300 Preliminary Design Review (PDR)
  - ❑ Software organization's Work Breakdown Structure (WBS) and Project relationship
  - ❑ Verification plan, incl. Mission-Level reviews, software milestone and peer reviews, and walkthroughs
  - ❑ Software-related Requests for Action (RFAs) and responses from the Code 300 PDR
  - ❑ Status of ICDs/IRDs and other external dependencies (documents, software, hardware, etc.)
  - ❑ Documentation plan, including each document's status and when it will be baselined
  - ❑ Product Assurance and Software Safety plans, activities, and status
  - ❑ Independent verification and validation (IV&V) plans and status
  - ❑ Infusion and reporting of Lessons Learned

- ❑ **Development overview,** highlighting changes since Code 300 PDR and current status
  - ❑ Requirements definition and management process, incl. documents used/produced, V&V, baselines
  - ❑ Design process, including standards used, documents produced, peer reviews conducted
  - ❑ Implementation process, incl. standards, review process, problem reporting, unit test and integration
  - ❑ Configuration Management (CM) processes, including discrepancy reporting and tracking (development and post-release)

- ❑ **Software overview,** highlighting changes since PDR
  - ❑ Overview of functional requirements and operations concepts
  - ❑ Major design changes since PDR, including changes to reuse
  - ❑ System architecture, external interfaces and end-to-end data flow
  - ❑ Software context diagram and overview of each software subsystem or major component
  - ❑ Failure detection and correction (FDC) requirements, approach, and detailed design
  - ❑ Software Requirements Verification Matrix (mapping requirements to subsystems/components)
  - ❑ Development environment (e.g., hardware diagram, operating system(s), compilers, debuggers, tools)

- ❑ **Detailed design –**
  For each subsystem or major component,
  - ❑ Functional and initialization requirements allocated to the subsystem/component, with safety-critical requirements highlighted
  - ❑ Requirements and design changes since PDR
  - ❑ Reused/heritage software from previous projects and modifications to that software
  - ❑ Subsystem/component context and design diagrams
  - ❑ Description of functionality and operational modes
  - ❑ Resource and utilization constraints (e.g., CPU, memory), including estimates of performance and how the software will adapt to changing margin constraints
  - ❑ Identification and formats of input and output data; data storage concepts and structures
  - ❑ Interrupts and/or exception handling, including event, FDC, and error messages
  - ❑ Current status and issues

Software Contents of the Mission-Level Critical Design Review (CDR)
(Continued)

- **Testing, delivery, and installation**
  - Charter and roles of the software test team
  - Documentation – titles and status of test plans, procedures, and traceability matrices
  - Test levels (e.g., unit testing, integration testing, system testing) – description, who executes, preparation and execution activities, test environment, standards followed, verification methods, and status
  - Build plan and test timeline, including a list of components and requirements to be tested in each build
  - Test environments for each test level –diagram and description of tools, testbeds, facilities
  - Software requirement verification recording, monitoring, and current status
  - System and acceptance testing – operational scenarios to be tested, incl. stress tests, recovery testing, and (if applicable) IT Security testing
  - Acceptance process – reviews, approval, and signoff processes
  - Delivery of source code and tools, version identification and documentation, installation of databases
- **Software status –** Current software size estimate; current schedule, milestone, staffing and cost/effort status
- **Risks –** with consequences and risk mitigation strategies
- **Issues, TBDs, and action items**